

Numériser le patrimoine I: standards et bonnes pratiques

Langages du web: HTML

Simon Gabay



HTML

HTML

HTML, pour *Hypertext Markup Language*, est un langage de balisage inventé en 1997 conçu pour représenter les pages web.

Plus d'informations sur [Wikipedia](#)

Si le XML est le fond, HTML est la forme.

Un document HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8"/>
    <meta name="keywords"
          content="key1, key2"/>
    <title>Titre</title>
  </head>
  <body>
    <h1>This is the h1</h1>
    <div>
      <h2>This is the h2</h2>
      <p>First paragraph.</p>
      <p>Another
        <br/>paragraph.</p>
    </div>
  </body>
</html>
```

HTML vs XML: points communs

- Structure arborescente
- Une balise ouverte doit être fermée
- Il est possible d'utiliser des balises autofermantes

Principaux éléments HTML

C'est un vocabulaire non-extensible. Il a quelques éléments importants:

- L'élément `` est un conteneur générique en ligne (*inline*) pour les contenus phrasés.

```
<div>Ici commence une div dont je veux <span>encadrer  
un passage</span> pour telle et telle raison</div>
```

- L'élément `` permet d'insérer une image. Le chemin vers le fichier est spécifié avec `src` et `alt` permet de fournir une description sommaire de l'image (si le navigateur n'arrive pas à l'ouvrir)

```

```

Principaux éléments HTML (suite)

- L'élément `` (*unordered list*) est une liste non numérotée

```
<ul>  
  <li>un item</li>  
  <li>un autre item</li>  
</ul>
```

On obtient une liste avec des puces :

- un item
- un autre item

Principaux éléments HTML (suite)

- L'élément `` (*ordered list*) fonctionne de la même manière

```
<ol>  
  <li>un item</li>  
  <li>un autre item</li>  
</ol>
```

On obtient une liste numérotée :

1. un item
2. un autre item

Principaux éléments HTML (suite)

- L'élément `<a>` permet de créer un lien. Il est utilisé avec `@href` pour encoder l'adresse du lien

```
<a href="www.unige.ch">uni de Genève</a>
```

On obtient un lien du type: [uni de Genève](#)

- L'élément `<meta>` permet d'ajouter des metadonnées au document

```
<meta charset="utf-8"/>  
<meta name="keywords"  
      content="key1, key2"/>
```

Deux attributs importants

- `@class` permet de catégoriser l'élément (utile pour javascript et CSS).

```
<div>Ici commence une div où j'identifie une ville comme  
  <span class="lieu">Paris</span> mais aussi beaucoup  
  d'autres comme <span class="lieu">Berne</span> ou bien  
  <span class="lieu">Berlin</span> parce que ça  
  m'intéresse.</div>
```

- `@id` permet d'identifier un élément (utile pour javascript et CSS).

```
<div>Ici commence une div dans laquelle je veux  
  identifier le nom <span id="simon">Simon</span> parce  
  que, encore une fois, ça m'intéresse.</div>
```

Exercice

Fabriquez votre CV en HTML

CSS

CSS

CSS (*Cascading Style Sheets*) est un langage informatique qui décrit la présentation des documents HTML (et aussi XML...)

Plus d'informations sur [Wikipedia](#)

Sélection

SELECTEUR	DECLARATION
sélecteur	{
	propriété1: valeur1;
	propriété2: valeur2;
	...
	propriétéN: valeurN;
	}

Syntaxe:

- Sélecteur au début
- Bloc de règles, délimité par les accolades `{ }`
- La propriété est séparée de la valeur par deux points `:`
- Une règle se termine par point-virgule `;`

Sélection

Un document HTML (ou XML)

```
<span id="Voltaire">Voltaire</span>  
<span class="auteur">Victor Hugo</span>  
<span class="poète">Baudelaire</span>
```

Lié à la règle CSS suivante

```
span{  
  font-style: italic;  
  font-weight: bold;  
}
```

Produit

Voltaire

Victor Hugo

Beaudelaire

Sélection II: précisions

Il est possible de sélectionner plusieurs éléments

```
span,div{  
  font-style: italic;  
  font-weight: bold;  
}
```

De restreindre la sélection via l'attribut

```
span[class]{  
  font-style: italic;  
  font-weight: bold;  
}
```

De restreindre la sélection via l'attribut et sa valeur

```
span[class="auteur"]{  
  font-style: italic;  
  font-weight: bold;  
}
```


Sélection III: raccourcis

Pour sélectionner tous les éléments portant une même classe, on utilise un point:

```
.auteur{  
    font-style: italic;  
}
```

Pour sélectionner l'élément portant un id précis, on utilise un dièse:

```
#Voltaire{  
    font-style: italic;  
}
```

Emplacement

1. *Inline*: à l'intérieur de l' `<élément>` avec l'attribut `@style`

```
<h1 style="color:blue;">Heading 1</h1>
```

2 *Interne*: avec l'élément `<style>` dans la balise `<head>`

```
<style>
  body {
    background-color: green;
  }
  h1 {
    color: maroon;
    margin-left: 40px;
  }
</style>
```

Emplacement

3. Externe

```
<!DOCTYPE html>
<html>
  <link rel="stylesheet" type="text/css" href="mon.css">
  <body>
    <h1>This is the h1</h1>
    <h2>This is the h2</h2>
    <p>My first paragraph.</p>
    <p>Another paragraph.</p>
  </body>
</html>
```

Contenu du fichier CSS

```
body {
  background-color: green;
}
h1 {
  color: maroon;
  margin-left: 40px;
}
```

Construire son dossier

```
racine
|-index.html
|-img
  |-img1.jpg
  |-img2.png
  |-img3.tif
|-css
  |-mon.css
|-scripts
  |-mon.js
|-docs
  |-unPdf.pdf
  |-fichierDocx.docx
|-README.md
```

Dans `<link rel="stylesheet" href="mon.css">`, la valeur de l'attribut `@href` n'est pas le nom du fichier `.css` mais le chemin vers le fichier `.css`, donc `mon.css`.

Quelques super-propriétés: **font**

- **font-size** pour la taille du texte

```
font-size: 16px; /* valeurs en px, Em, % */
```

- **font-weight** pour les gras

```
font-weight: bold; /* valeurs: bold, bolder, lighter */
```

- **font-style** pour les italiques

```
font-style: italic; /* valeurs: italic, oblique */
```

- **font-variant** pour les capitales

```
font-variant: small-caps;
```

Quelques super-propriétés: **text**

- **text-indent** pour l'alinéa

```
text-indent:16px; /* valeurs en px, Em, % */
```





- **text-align** pour l'alignement

```
text-align:justify; /* valeurs: left, center, right*/
```

- **text-decoration** ligne au dessus, au dessous, dessus

```
text-align:decoration; /* valeurs: underline, overline,  
line-through*/
```

Quelques super-propriétés: **background**

- **background-color** pour la couleur de fond. On utilise directement le nom de la couleur ou son code ( #000000 #000000,  #FF0000 #FF0000,  #0000FF #0000FF,  #FFFF00 #FFFF00 ... cf. liste sur le [site du W3C](#))

```
background-color:green; /* nom ou code couleur*/
```

- **background-align** pour l'image de fond

```
background-image:url(images/image_de_fond.png);
```

Quelques super-propriétés: **margin** et **padding**

- **padding** pour la marge intérieure (entre le contenu et la bordure)

```
padding:16px; /* valeurs en px, Em, % */
```

- Il est possible de préciser quelle marge (haut, bas, gauche...)

```
padding-top:16px; /* valeurs en px, Em, % */  
padding-left:16px;
```

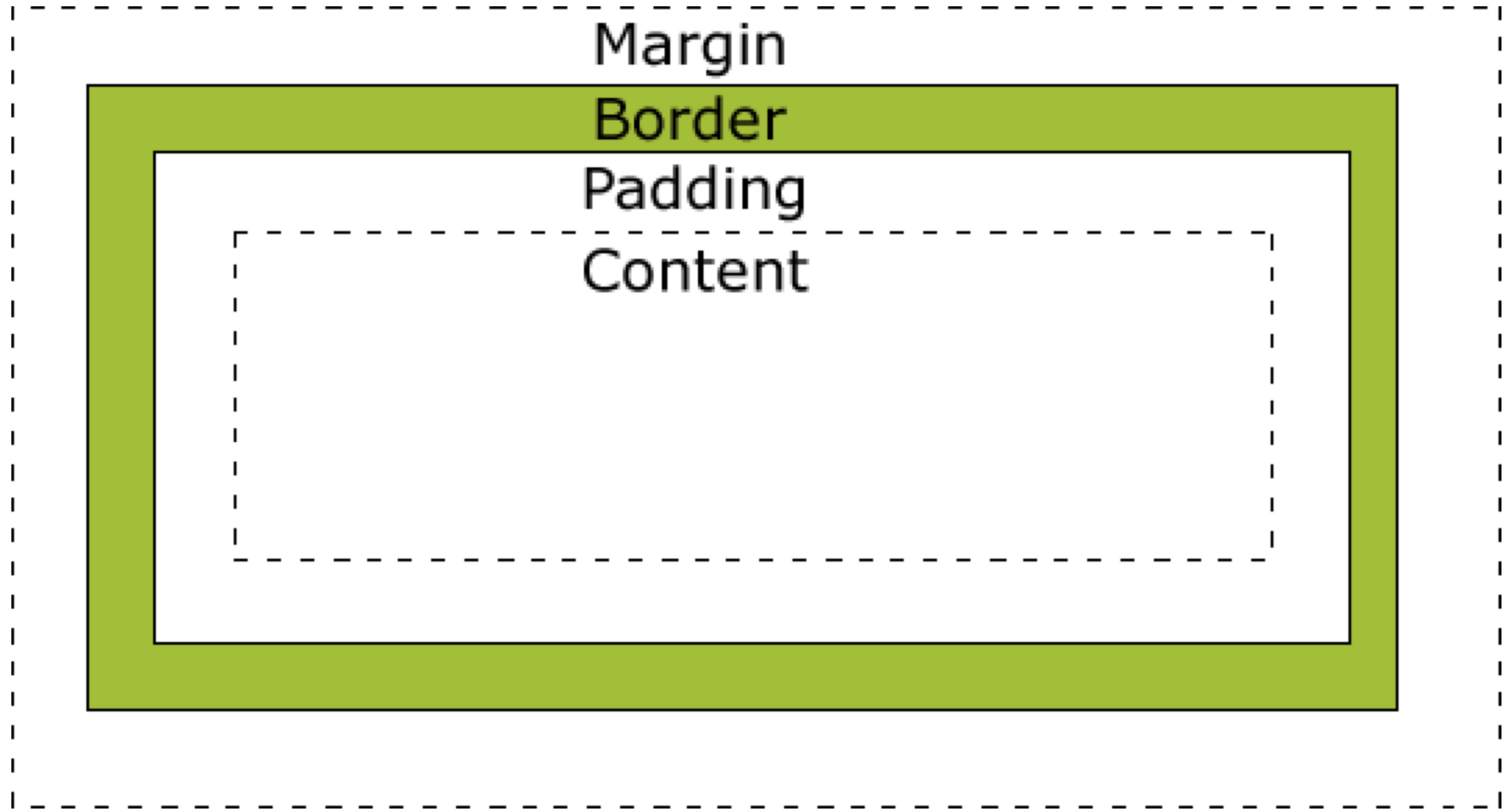
- **margin** pour la marge extérieure (après la bordure)

```
margin:16px; /* valeurs en px, Em, % */
```

- Il est possible de préciser quelle marge (haut, bas, gauche...)

```
margin-top:16px; /* valeurs en px, Em, % */  
margin-left:16px;
```


Margin vs padding



À savoir

- `color` permet de changer la couleur de la police ([cf. W3C](#))

```
color:red; /* nom ou code couleur*/
```

- `display` définit le type d'affichage utilisée pour le rendu d'un élément ([cf. W3C](#))

```
display:block; /* nom ou code couleur*/
```

- Il est possible (et même conseillé) de commenter son code

```
/* commentaire */
```

Conflits

En cas de conflit, la dernière règle s'applique

```
div {  
    background-color: green;  
}  
div[class="chapter"] {  
    background-color: yellow;  
}
```

→ Toutes les `<div>` auront un fond vert, sauf celle avec un attribut `@type="chapter"`

```
div[class="chapter"] {  
    background-color: yellow;  
}  
div {  
    background-color: green;  
}
```

→ Toutes les `<div>` auront un fond vert, même celle avec un attribut `@type="chapter"`